REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
02-05-2013	Final Report	1-Jun-2009 - 31-Oct-2012

4. TITLE AND SUBTITLE

On the Use of Software Metrics as a Predictor of Software Security Problems

5a. CONTRACT NUMBER
W911NF-09-1-0239

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
611102

6. AUTHORS
Laurie Williams

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES
North Carolina State University
Research Administration
2701 Sullivan Drive, Suite 240
Raleigh, NC 27695 -7514

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
53488-CS.7

12. DISTRIBUTION AVAILIBILITY STATEMENT
Approved for Public Release; Distribution Unlimited

13. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT
Relying on one validation and verification (V&V) alone cannot detect all of the security problems of a software system. Each class of V&V effort detects different class(s) of faults in software. Even composing a series of V&V efforts, one can never be completely sure that all faults have been detected. Additionally, security-related V&V efforts must continuously be updated to handle the newest forms of exploits of underlying vulnerabilities in software. The alerts produced by automated static analysis (ASA) tools and other static metrics have been shown

15. SUBJECT TERMS
security, software, metrics

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dennis Kekas
UU	UU	UU	UU		19b. TELEPHONE NUMBER 919-515-5297

Standard Form 298 (Rev 8/98)
Prescribed by ANSI Std. Z39.18

Report Title

On the Use of Software Metrics as a Predictor of Software Security Problems

ABSTRACT

Relying on one validation and verification (V&V) alone cannot detect all of the security problems of a software system. Each class of V&V effort detects different class(s) of faults in software. Even composing a series of V&V efforts, one can never be completely sure that all faults have been detected. Additionally, security-related V&V efforts must continuously be updated to handle the newest forms of exploits of underlying vulnerabilities in software. The alerts produced by automated static analysis (ASA) tools and other static metrics have been shown to be an effective estimator of the actual reliability in a software system. Predictions of defect density and high-risk components can be identified using static analyzers early in the development phase. Our research hypothesis is the actual number of security vulnerabilities in a software system can be predicted based upon the number of security-related alerts reported by one or more static analyzers and by other static metrics. We built and evaluated statistical prediction model are used to predict the actual overall security of a system.

Enter List of papers submitted or published that acknowledge ARO support from the start of the project to the date of this printing. List the papers, including journal references, in the following categories:

(a) Papers published in peer-reviewed journals (N/A for none)

Received	Paper

TOTAL:

Number of Papers published in peer-reviewed journals:

(b) Papers published in non-peer-reviewed journals (N/A for none)

Received	Paper

TOTAL:

Number of Papers published in non peer-reviewed journals:

(c) Presentations

Number of Presentations: 0.00

Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received	Paper

TOTAL:

Number of Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received	Paper	
05/01/2013	1.00	Michael Gegick, Peter Rotella, Laurie Williams . Predicting Attack-Prone Components, International Conference on Software Testing, Verification, and Validation (ICST) 2009. 2009/04/01 00:00:00, . : ,
05/01/2013	3.00	Laurie Williams, Michael Gegick, Andrew Meneeley. Protection Poker: Structuring Software Security Risk Assessment and Knowledge Transfer, International Symposium on Engineering Secure Software and Systems (ESSoS) 2009. 2009/02/04 00:00:00, . : ,
05/01/2013	4.00	Michael Gegick, Pete Rotella, Laurie Williams. Toward Non-Security Failures as a Predictor of Security Faults and Failures,, International Symposium on Engineering Secure Software and Systems (ESSoS) 2009. 2009/02/04 00:00:00, . : ,
05/01/2013	5.00	Michael Gegick, Laurie Williams. Correlating Automated Static Analysis Alert Density to Reported Vulnerabilities in Sendmail, Metricon. 2007/08/07 00:00:00, . : ,
05/01/2013	6.00	Michael Gegick, Laurie Williams, Jason Osborne, Mladen Vouk . Prioritizing Software Security Fortification through Code-Level Security Metrics, Quality of Protection Workshop at the ACM Conference on Computers and Communications Security (CCS) 2008. 2008/10/27 00:00:00, . : ,
TOTAL:	**5**	

Number of Peer-Reviewed Conference Proceeding publications (other than abstracts):

(d) Manuscripts

Received	Paper

TOTAL:

Number of Manuscripts:

Books

Received	Paper

TOTAL:

Patents Submitted

Patents Awarded

Awards

Graduate Students

NAME	PERCENT_SUPPORTED	Discipline
Michael Gegick	1.00	
John Slankas	0.50	
FTE Equivalent:	**1.50**	
Total Number:	**2**	

Names of Post Doctorates

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Names of Faculty Supported

NAME	PERCENT_SUPPORTED	National Academy Member
Laurie Williams	0.05	
FTE Equivalent:	**0.05**	
Total Number:	**1**	

Names of Under Graduate students supported

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Student Metrics

This section only applies to graduating undergraduates supported by this agreement in this reporting period

The number of undergraduates funded by this agreement who graduated during this period:	0.00
The number of undergraduates funded by this agreement who graduated during this period with a degree in science, mathematics, engineering, or technology fields:	0.00
The number of undergraduates funded by your agreement who graduated during this period and will continue to pursue a graduate or Ph.D. degree in science, mathematics, engineering, or technology fields:	0.00
Number of graduating undergraduates who achieved a 3.5 GPA to 4.0 (4.0 max scale):	0.00
Number of graduating undergraduates funded by a DoD funded Center of Excellence grant for Education, Research and Engineering:	0.00
The number of undergraduates funded by your agreement who graduated during this period and intend to work for the Department of Defense	0.00
The number of undergraduates funded by your agreement who graduated during this period and will receive scholarships or fellowships for further studies in science, mathematics, engineering or technology fields:	0.00

Names of Personnel receiving masters degrees

NAME
Total Number:

Names of personnel receiving PHDs

NAME	
Michael Gegick	
Total Number:	**1**

Names of other research staff

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Sub Contractors (DD882)

Inventions (DD882)

Scientific Progress

Technology Transfer

Predicting Attack-prone Components with Source Code Static Analyzers

Statement of Problem Studied

No single vulnerability detection technique can identify all vulnerabilities in a software system. However, the vulnerabilities that are identified from a detection technique may be predictive of the residuals. We focus on creating and evaluating statistical models that predict the components that contain the highest risk residual vulnerabilities.

The cost to find and fix faults grows with time in the software life cycle (SLC). A challenge with our statistical models is to make the predictions available early in the SLC to afford for cost-effective fortifications. Source code static analyzers (SCSA) are available during coding phase and are also capable of detecting code-level vulnerabilities. We use the code-level vulnerabilities identified by these tools to predict the presence of additional coding vulnerabilities and vulnerabilities associated with the design and operation of the software. *The goal of this research is to reduce vulnerabilities from escaping into the field by incorporating source code static analysis warnings into statistical models that predict which components are most susceptible to attack*.

The independent variable for our statistical model is the count of security-related source SCSA warnings. We also include the following metrics as independent variables in our models to determine if additional metrics are required to increase the accuracy of the model: non-security SCSA warnings, code churn and size, the count of faults found manually during development, and the measure of coupling between components. The dependent variable is the count of vulnerabilities reported by testing and those found in the field.

Summary of Most Important Results

We evaluated our model on three commercial telecommunications software systems. Two case studies were performed at an anonymous vendor and the third case study was performed at Cisco Systems. Each system is a different technology and consists of over one million source lines of C/C++ code. The results show positive and statistically significant correlations between the metrics and vulnerability counts. Additionally, the predictive models produce accurate probability rankings that indicate which components are most susceptible to attack. The models are evaluated with receiver operating characteristic curves where each case study showed over 92% of the area was under the curve. We also performed five-fold cross-validation to further demonstrate statistical confidence in the models. Based on these results we contribute the following theory:

Theory: Above a statistically determined threshold, SCSA vulnerability warnings are in the same components as vulnerabilities that are likely to be exploited.

Components that contain security-related warnings identified by SCSA are also likely to contain other exploitable vulnerabilities. Software engineers should systematically inspect and test code for other vulnerabilities when a security-related warning is present. Fortifying these vulnerabilities may facilitate other techniques to identify more undetected vulnerabilities.